

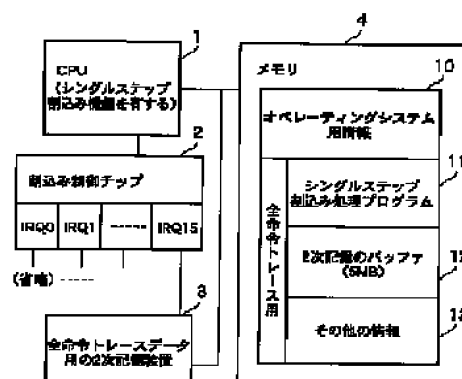


PATENT ABSTRACTS OF JAPAN

(11) Publication number: **2000066923 A**(43) Date of publication of application: **03.03.00**(51) Int. Cl. **G06F 11/28**(21) Application number: **10239249**(22) Date of filing: **25.08.98**(71) Applicant: **NIPPON TELEGR & TELEPH
CORP <NTT>**(72) Inventor: **SUGIMURA YASUSHI****(54) METHOD FOR CONTROLLING
ALL-INSTRUCTION TRACING IN EMULATOR****(57) Abstract:**

PROBLEM TO BE SOLVED: To realize an emulator all-instruction tracing controlling method capable of correctly and accurately executing the tracing of all instructions in an emulator which can not be realized by conventional processing.

SOLUTION: Even when the emulator tries to turn off the trace flag of a CPU 1 during the execution of all-instruction tracing, an all-instruction tracing single step interruption processing program prevents the trial, collects all-instruction tracing data and even when the emulator turns on the tracing flag of the CPU 1, the program turns off the flag and does not collect the all-instruction tracing data.



COPYRIGHT: (C)2000,JPO

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号
特開2000-66923
(P2000-66923A)

(43) 公開日 平成12年3月3日(2000.3.3)

(51) Int.Cl. ⁷	識別記号	F I	テーマコード* (参考)
G 0 6 F 11/28	3 1 0	G 0 6 F 11/28	3 1 0 B 5 B 0 4 2

審査請求 未請求 請求項の数1 O L (全 8 頁)

(21) 出願番号 特願平10-239249

(22) 出願日 平成10年8月25日(1998.8.25)

(71) 出願人 000004226

日本電信電話株式会社
東京都千代田区大手町二丁目3番1号

(72) 発明者 杉村 康

東京都新宿区西新宿三丁目19番2号 日本
電信電話株式会社内

(74) 代理人 100083806

弁理士 三好 秀和 (外1名)

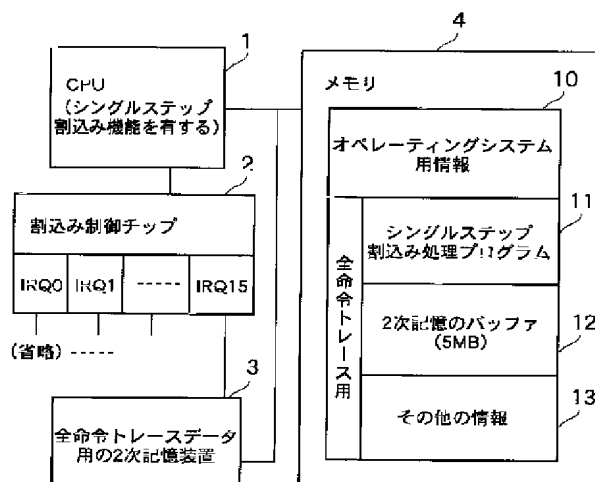
Fターム(参考) 5B042 GA03 HH23 HH30 JJ40

(54) 【発明の名称】 エミュレータにおける全命令トレースの制御方法

(57) 【要約】

【課題】 従来の処理では実現が不可能であったエミュレータにおける全命令トレースをすべて正しく適確に行うことができるエミュレータにおける全命令トレースの制御方法を提供する。

【解決手段】 全命令トレース実行中時にエミュレータがCPUのトレースフラグをオフにしようとしても、全命令トレース用シングルステップ割り込み処理プログラムがそれを阻止し、全命令トレースデータの収集を行い、また全命令トレース仮停止中時にエミュレータがCPUのトレースフラグをオンにしても、全命令トレース用シングルステップ割り込み処理プログラムがオフに戻し、この時全命令トレースデータの収集を行わない。



【特許請求の範囲】

【請求項1】 情報処理装置内の中央演算ユニットが全命令トレースを行い、このトレース対象のオペレーティングシステムが他のシステムのシステムコールのエミュレーションを行い、このエミュレーションを含むすべての命令の実行の全命令トレースを行う場合において、全命令トレース用シングルステップ割り込み処理プログラムの外部変数の計数手段としてカウンタを設け、全命令トレースの実行中においては、前記プログラムがエミュレーションにおけるトレースフラグをオフにする命令を検出した場合、このオフを阻止し、前記カウンタの値に1を加算し、全命令トレースの終了以降においては、前記カウンタの値が0でない場合に、前記プログラムがシングルステップ割り込みの発生を監視し、該割り込みが発生した場合は、トレースフラグをオフとし、前記カウンタから1を減算し、この結果のカウンタの値が0でない場合は、前記監視を継続し、この結果のカウンタの値が0である場合は、前記監視を停止して、全命令トレースのすべての動作を終了することを特徴とするエミュレータにおける全命令トレースの制御方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、情報処理装置内の中央演算ユニット（以下、CPUと略記する）が全命令トレースを行い、このトレース対象のオペレーティングシステムが他のシステムのシステムコールの動作をエミュレートするエミュレータにおける全命令トレースの制御方法に関し、特に電子計算機、ワードプロセッサ、電子交換機、ワークステーション等の情報処理装置内のCPUの全命令トレースの制御方法に関する。

【0002】

【従来の技術】一般に、性能評価等の目的のため、評価対象のオペレーティングシステムのプログラム（該オペレーティングシステム上で動作するすべてのアプリケーションプログラムを含み、全命令トレースを行うためのプログラムを除く、以下省略）の命令群の内、CPUが実行するすべての命令に関する情報（実行された命令の内容、命令の長さ、レジスタの変化状況、変化したレジスタの内容、その他必要に応じてプログラムの管理単位であるスレッドの情報等の情報）を1命令の実行直前毎に収集することを、全命令トレースと言い、その全命令トレースによって収集された情報を全命令トレースデータと言う。上記プログラムは、関数（プログラムからコールされ、入り口が1つであるプログラム）並びに割り込み処理プログラム（割り込み等を契機としてコールされるプログラムであり、通常はプログラムからはコールされないプログラム）からなり、全命令トレースにおいてはそれらのすべてがトレース対象である（関数のみのトレースしか行わないものは、全命令トレースとは言わない）。

【0003】シングルステップ割り込み機能を持つCPUの元での全命令トレースにおいては、関数に関しては予めCPUのトレースフラグをONとしておき、割り込み処理に関しては、その先頭でCPUのトレースフラグをONとする等によって、CPUに1命令の実行直前毎にシングルステップ割り込みを起こさせることが一般的に行われている。

【0004】CPUが評価対象のオペレーティングシステムの1命令を実行する直前毎に、CPU自身に起こさせるシングルステップ割り込みを利用して、CPUは全命令トレース用シングルステップ割り込み処理プログラムに制御を渡し、そのプログラムが全命令トレースデータを収集し、一旦全命令トレースデータバッファというメモリ上のエリアに格納して、上記割り込み元にリターンし、リターン後にCPUは、割り込まれた命令を1命令実行し、その実行が終わって、次の1命令を実行する直前に、上記と同様にシングルステップ割り込みを起こさせることにより、評価対象のオペレーティングシステムのプログラムの命令群の内、実行されるすべての命令の全命令トレースが行われる。

【0005】上記全命令トレースにおいては、一般に、何らかの手法で全命令トレースの開始点を検出した時点（その手法は多数あり、任意、以下全命令トレース開始時点と略記）で、CPUのトレースフラグをONとし、何らかの手法で全命令トレースの停止点を検出した時点（その手法は多数あり、任意、以下全命令トレース停止時点と略記）でCPUのトレースフラグをOFFとすることにより、その時実行中のプログラムの全命令トレースを制御するが、以下に示す手法の中で示すように、それだけでは全命令トレース開始時点～全命令トレース終了時点までのすべてのスレッドや割り込み処理の全命令トレースを行うことができないので、従来より、一般に全命令トレースにおいては下記の2つの手法を併用する。

【0006】（1）全命令トレースは、割り込み処理においても、全命令トレースデータを収集しなければならないが、割り込み処理においては、一般に、CPUが、割り込まれた時のトレースフラグをスタックに退避し、トレースフラグをOFFとした後に、割り込み処理プログラムに制御を渡す。従って、何もしないと、割り込み処理プログラムの命令の実行においてシングルステップ割り込みが発生しないため、全命令トレースデータを収集することができない。それに対処するために、上記全命令トレース開始時点で、外部変数「全命令トレース実行中」に1（初期値は0）を設定し、上記全命令トレース停止時点で、該外部変数の値を0に戻し、割り込み処理の先頭で、該外部変数の値をチェックし、該値が1の時のみ、CPUのトレースフラグに1を設定することにより、全命令トレース実行中にのみ、割り込み処理の部分も全命令トレースを行う（当然ながら、シングルステ

ップ割り込みは、全命令トレースの情報を収集する処理であるので、上記処理は行わない。シングルステップ割り込み処理だけは、全命令トレース対象外である。

【0007】(2)一般にオペレーティングシステム内にはスレッド(独立して走行するプログラムの単位)が多数存在し、オペレーティングシステムは、それらのスレッドを適宜切り換えて実行することにより、CPUの有効利用を図る。しかし、CPUが1時点で実行しているスレッドは、ただ1つである。そのため、その時実行されていないスレッドは、その実行は一旦中断されている。そして、その中断されたスレッドを適宜再開するために、中断時点のレジスタ等の情報を、そのスレッドのスタックエリアに退避している。その中には、CPUのトレースフラグも含まれる。従って、何もしないと、現在のスレッドの実行が中断され、既に中断されているスレッドを再開する時に、スタックエリアに退避されているトレースフラグ等は、中断直前の状態に回復され、その時に、トレースフラグがOFFになってしまうため、全命令トレースが行われない結果となる。それらを回避する(すべてのスレッドの全命令トレースを実行させる)ために、一般に、上記トレース開始時点において、すべてのスレッドのスタックエリアに既に退避されているトレースフラグをすべてONにし、上記トレース停止時点において、OFFに戻すことにより、全命令トレース実行中に、すべてのスレッドの全命令トレースが行われるようにしている。上記スタックエリアは、一般にカーネルスタックエリアと呼ばれ、そのスタックエリアは、カーネル空間(全命令トレース用シングルステップ割り込み処理プログラム等のカーネル関連プログラムから可視である空間)にあり、スレッド毎に予め決められた構造体内にあるので、上記のように、トレース開始時点とトレース停止時点に、一度に変更することが可能である。

【0008】

【発明が解決しようとする課題】しかしながら、最近においては、1つのオペレーティングシステム内において別のオペレーティングシステムのシステムコールの動作を、ユーザ空間上でエミュレートするエミュレータが実現されている。該エミュレータは、オペレーティングシステムが動作するカーネル空間で動作するのではなく、スレッド等毎にあるユーザ空間上で動作し、そのため、そのスタック情報は、ユーザスタック(上記カーネルスタックとは別のユーザ毎のスタック)に格納される。

【0009】一方、他のオペレーティングシステムのシステムコールをユーザ空間上でエミュレーションするためには、システムコールと呼ばれるプログラムはシステム空間になければならないので、システムコールで一旦システム空間に行った時に、そのシステムコールルーチン等によりスタック等に退避された情報を、ユーザスタック上にコピーし、その後、ユーザ空間にジャンプし

て、システムコールのエミュレート動作を行い、最後に、^①上記ユーザスタック上にコピーされた情報を回復することによって、上記システムコールの発行元にリターンする。そしてその回復される情報の中に、トレースフラグが含まれる。即ち、そのトレースフラグの情報は、ユーザスタック内にあり、ユーザスタック内のどの位置にあるかは、そのスレッドでどのようにユーザスタックが使用されたかに依存し(上記のカーネルスタックのように、スレッド毎の特定の位置にある訳ではないので)、従来の技術で述べた手法のように、トレース開始時点やトレース終了時点では、上記のトレースフラグのアドレスを特定することができないので、変更することができない(そのスタックを実際に退避回復する場所では、その位置を知ることができない)。

【0010】従って、従来の手法においては、別のオペレーティングシステムのシステムコールの動作をユーザ空間上でエミュレートするエミュレータが搭載されているシステムにおいては、上記の^②のトレースフラグの回復が行われた時点以降はトレースフラグがOFFになってしまい、そのため、そのエミュレータが動作しているスレッドの全命令トレースを行うことができないという問題がある。

【0011】本発明は、上記に鑑みてなされたもので、その目的とするところは、従来の処理では実現が不可能であったエミュレータにおける全命令トレースをすべて正しく適確に行うことができるエミュレータにおける全命令トレースの制御方法を提供することにある。

【0012】

【課題を解決するための手段】上記目的を達成するため、請求項1記載の本発明は、情報処理装置内の中央演算ユニットが全命令トレースを行い、このトレース対象のオペレーティングシステムが他のシステムのシステムコールのエミュレーションを行い、このエミュレーションを含むすべての命令の実行の全命令トレースを行う場合において、全命令トレース用シングルステップ割り込み処理プログラムの外部変数の計数手段としてカウンタを設け、全命令トレースの実行中においては、前記プログラムがエミュレーションにおけるトレースフラグをオフにする命令を検出した場合、このオフを阻止し、前記カウンタの値に1を加算し、全命令トレースの終了以降(以下、全命令トレース仮停止中と略記)においては、前記カウンタの値が0でない場合に、前記プログラムがシングルステップ割り込みの発生を監視し、該割り込みが発生した場合は、トレースフラグをオフとし、前記カウンタから1を減算し、この結果のカウンタの値が0でない場合は、前記監視を継続し、この結果のカウンタの値が0である場合は、前記監視を停止して、全命令トレースのすべての動作を終了することを要旨とする。

【0013】請求項1記載の本発明にあつては、全命令トレース実行中にエミュレータがCPUのトレースフ

ラグをオフにしようとしても、全命令トレース用シングルスステップ割り込み処理プログラムがそれを阻止し、全命令トレースデータの収集を行い、また全命令トレース仮停止中にエミュレータがCPUのトレースフラグをオンにしても、全命令トレース用シングルスステップ割り込み処理プログラムがオフに戻し、この時全命令トレースデータの収集を行わないため、エミュレータにおける全命令トレースを適確に行うことができる。

【0014】

【発明の実施の形態】本発明の実施形態を説明する前に、まず上述したように、従来、エミュレータが動作しているスレッドの全命令トレースを行うことができないという問題を解決するために利用する4つの事実および本発明の概要について説明する。この4つの事実は以下の通りである。

【0015】^① 上記トレースフラグの回復は、POPFDという特定の命令によってのみ可能であり、エミュレータは上記命令によりトレースフラグを回復する。

【0016】^② 上記の特定の命令においては、トレースフラグに設定される値は、(命令仕様に基づいて)CPUのスタックポインタが示すアドレスの位置に存在する。

【0017】^③ 上記^②のスタックポインタの内容は、上記命令を実行する直前でシングルスステップの割り込みが発生した場合、CPUの機能により、スタック内に格納される。

【0018】^④ 従来の手法においては、上記の特定の命令の実行の直前のシングルスステップ割り込みまでは、全命令トレースが正常に行われる(上記命令の次の命令以降からシングルスステップ割り込みが起こらなくなる)。

【0019】上記の事実にしたがって、全命令トレース開始時点から全命令トレース仮停止時点の間(以下、全命令トレース実行中時と略記)において以下を行う。

【0020】(1) 命令が実行される直前に発生するシングルスステップ割り込みによって制御が渡される「全命令トレース用シングルスステップ割り込み処理プログラム」は、全命令トレース実行中時においては、その時に実行されようとした命令が、上記POPFDであるかどうかをチェックし、上記の特定の命令でないなら、従来と同じ処理を行う。上記の特定の命令であれば、従来の処理を行う前に、以下を実施する。

【0021】(A) 「CPUによって退避されたスタックポインタ」の内容が示すアドレスのエリアの内容(即ちトレースフラグが格納されているエリアの内容)をチェックし、

(a) トレースフラグがONであるなら、そのエリアの内容のトレースフラグをONとする。これによって、全命令トレース用シングルスステップ割り込み処理プログラムからリターンした時に実行されるPOPFDによってCPUのトレースフラグがONに設定される。

(b) トレースフラグがOFFであるなら、何もしない。以上によって、エミュレータを含むすべての全命令トレースが可能となる。

【0022】しかしながら、上記の実現だけでは、次の問題が発生する。

【0023】上記(A)によってONが設定されたトレースフラグは、エミュレータによってスタックに退避される場合があるので、全命令トレース停止時に従来の方法によって、すべてのスレッドのトレースフラグをOFFにしても、その後、上記で退避されていたトレースフラグが回復された時点で、再び、シングルスステップ割り込みが発生してしまい、何もしないと再び全命令トレースが実施されてしまう。

【0024】それらに対処するために、以下を行う。

【0025】(2) 外部変化「トレースON化累積カウンタ」「全命令トレース仮停止」を設け、予め0に初期設定しておく。

【0026】(3) 上記(1)(A)(a)で、「トレースON化累積カウンタ」に1を加算する。

【0027】(4) 全命令トレース仮停止時に、「トレースON化累積カウンタ」が0であるかどうかをチェックし、0でないなら、「全命令トレース実行中」に0を設定して、従来の全命令トレースの停止を行うだけでなく、「全命令トレース仮停止」に1を設定する。

【0028】0なら、「全命令トレース実行中」に0を設定して、従来の全命令トレースの停止を行う。

【0029】(5) 全命令トレース用シングルスステップ割り込み処理プログラムは、「全命令トレース仮停止」が1であるかどうかをチェックし、0であるなら、従来の処理と、上記(1)の処理を行うが、1である場合は、従来の処理も、上記(1)も行わず(全命令トレースデータの収集等は行わず)以下を行う。

【0030】(a) スタックエリアに退避されているエラーフラグの内容をOFFとすることにより、全命令トレース用シングルスステップ割り込み処理プログラムからリターンした時に、CPUのトレースフラグをOFFとさせる。

(b) 「トレースON化累積カウンタ」より1を減算する。

(c) 上記(b)の結果が0であるなら、「全命令トレース仮停止」に0を設定することにより、全命令トレースが完全に停止する。

(d) 全命令トレース用シングルスステップ割り込み処理プログラムよりリターンする。

【0031】以上により、オペレーティングシステム内に別のオペレーティングシステムのシステムコールの動作をユーザ空間上でエミュレートするエミュレータが搭載されていても、エミュレータの動作を含んだ全命令トレースを正しく行うことが可能となる。

【0032】本発明においては、上述したように、オペ

レーティングシステム内に別のオペレーティングシステムのシステムコールの動作をユーザ空間上でエミュレートするエミュレータが搭載されている場合に、全命令トレース実行中時の場合は、エミュレータがトレースフラグをOFFにしようとしても、それを実施する命令の実行の直前で、全命令トレース用シングルスステップ割り込み処理プログラムがそれを阻止し、また全命令トレース仮停止時の後に、エミュレータがトレースフラグをONにしても、そのONによる（次の命令が実行される直前に発生する）シングルスステップ割り込みによる全命令トレース用シングルスステップ割り込み処理プログラムの走行で、全命令トレースデータを収集せず、且つ、該割り込み処理プログラムからリターンした時に、トレースフラグがOFFになるように、スタックの内容を変更するので、エミュレータの動作を含んだ全命令トレースを正しく行うことが可能となる。

【0033】次に、図面を用いて、本発明の一実施形態に係るエミュレータにおける全命令トレースの制御方法について説明する。図1は、本発明の一実施形態に係るエミュレータにおける全命令トレースの制御方法を実施する装置の構成を示すブロック図である。同図に示す実施形態は、シングルスステップ割り込み機能を有するCPU1（例えば、INTEL Pentium）、割り込み制御チップ2（例えば、INTEL 8259）、全命令トレースデータ用の2次記憶装置3（例えば、IDEセカンダリーHD、IRQ番号15配下）、およびメモリ4を有する。メモリ4は、オペレーティングシステム用情報10、全命令トレース用シングルスステップ割り込み処理プログラム11、全命令トレース用2次記憶装置のバッファ12、および全命令トレース用のその他の情報13を記憶している。

【0034】まず、シングルスステップ割り込み機能を有するCPU1は、割り込み制御チップ2と信号線によって接続されており、該チップからの割り込みの有無や該チップの状態の情報等を受け取ることができ、また、該チップへの割り込み受け取り完了の通知等の情報を出力できる。また、CPU1は、全命令トレースデータ用の2次記憶装置3およびメモリ4とバスによって接続されており、それらの装置並びにメモリに対する情報のリードおよびライトが可能である。メモリ4内の情報は、オペレーティングシステム用情報10、全命令トレース用シングルスステップ割り込み処理プログラム11、2次記憶装置のバッファ12、およびその他の情報13より構成される。

【0035】図2は、オペレーティングシステム用情報10の中にある全命令トレース用シングルスステップ割り込み処理プログラム11の走行時のスタックの構造であり、シングルスステップ割り込みによって作成されるスタック50と、その割り込みが発生する直前のユーザの状態フラグ60、並びに、スタックポインタ70からなる。

また、スタック50は、旧スタックセグメント番号51、旧スタックポインタ52、旧状態フラグ53、旧コードセグメント番号54、旧命令アドレス55、レジスタ情報群56からなる。

【0036】それらの内、スタックポインタ70は、レジスタ情報群56の最後の4バイトのアドレスをポイントしており、旧スタックポインタ52は、ユーザの状態フラグ60をポイントしている関係を示す。

【0037】図3は、状態フラグ90の構造を示しており、該状態フラグは、32ビットの情報からなり、その内8番目のビットがトレースフラグ91であることを示す。この状態フラグ90の構造は、ユーザの状態フラグ60および旧状態フラグ53の中の構造を示す。

【0038】次に、図4および図5に示すフローチャートを参照して、上記実施形態の作用について説明する。なお、この説明では、全命令トレース実行中時においてエミュレータが初めてPOPF命令を実行しようとしている時を例にとって説明する。

【0039】この時、外部変数「全命令トレース実行中」の値は1、「トレースON化累積カウンタ」および「全命令トレース仮停止」の値は0である（外部変数とは、どこからでも見える領域であり、構成上はどこに存在してもよい。本実施形態においては、図1のブロック13で示すその他の情報の中にある）。

【0040】全命令トレース実行中時において、エミュレータがPOPF命令を実行しようすると、その命令の実行の直前に、シングルスステップ割り込みが発生し、CPU1は、図4の全命令トレース用シングルスステップ割り込み処理プログラム11のステップS100に制御を渡す。その時、スタック50には、図2の51～55の情報、CPU1のスタックポインタには旧命令アドレス55のエリアのアドレスを、CPU1は、自動的に設定する。次に、CPU1は図4のレジスタ等の退避（ステップS100）を実行すると、スタック50の中のレジスタ情報群56にレジスタ等の内容を退避し、CPU1のスタックポインタ70の内容は、図2に示すように、レジスタ情報群56の最後の4バイトの位置のアドレスとなる。

【0041】次に、外部変数「全命令トレース仮停止」の値が0かどうかをチェックし（ステップS101）、ここではYESであるので、ステップS102へ行き、外部変数「全命令トレース実行中」が0でないかどうかをチェックする（ステップS102）。ここでは、全命令トレース実行中であり、YESであるので、全命令トレースデータを収集してバッファに格納し、適宜、2次記憶装置3に出力する（ステップS104）。それから、シングルスステップ割り込みを起こした命令のアドレスがユーザ空間であるかどうかを「スタック50内の旧命令アドレス（この場合、上記POPF命令のアドレス）がユーザ空間（本実施形態では、ユーザ空間のアド

レスは値が0XC0000000より小)であるか] チェックする(ステップS105)。該命令はエミュレータの命令であるから、ユーザ空間であるので、YESとなり、次に、該命令がPOPF Dの命令であるかどうかを「スタック50内の旧命令アドレス55を当プログラムが動作しているシステム空間の値に変換して、そのアドレスがポイントするエリアの内容が、0X9D (POPF Dの命令コード)であるかを」チェックし(ステップS106)、この場合YESであるので、ステップS107で、旧スタックポインタ52の内容よりユーザの状態フラグ60のアドレスを得、状態フラグ60の内、図3のトレースフラグ91を得、その値がOFFであるかどうかをチェックする(ステップS108)。ここでは、全命令トレース実行中時でエミュレータが初めてPOPF Dを実行しようとしている時であるので、該ビットはOFFであり、YESとなり、ステップS109へ行き、ユーザの状態フラグ60のトレースフラグ91に1(ON)を設定し、外部変数「トレースON化累積カウンタ」に1を加算(結果は1となる)した後、ステップS200へ進む。

【0042】ステップS200では、(一般にオペレーティングシステム用情報10内の現スレッドアドレスと、予めユーザが指定した全命令トレースの終了用スレッドアドレスとが一致し、且つスタック50内の旧命令アドレス55と、予めユーザが指定した全命令トレースの終了用命令アドレスとが一致するかどうかをチェックすることにより)全命令トレースの終了が必要かどうかをチェックし、(エミュレータ内でトレースを終了することは一般にないので)この場合NOであるので、ステップS205へ行き、レジスタ等の回復を行い、ステップS206で、IRET命令で割り込み元に戻る。

【0043】割り込み元は、上記POPF Dの命令であるから、その命令が実行されるが、そのときユーザの状態フラグ60のトレースフラグ91は、ステップS109でONに変更されているので、上記命令の実行により、CPUのトレースフラグはONになり、次の命令の実行直前に降も、シングルステップ割り込みは継続して発生することとなる。

【0044】次に、全命令トレースの終了が必要な命令の実行の直前に発生するシングルステップ割り込みの場合を考える。上記と同様に全命令トレース用シングルステップ割り込み処理プログラムに制御を渡されると、レジスタ等の退避を行い(ステップS100)、全命令トレース仮停止=0かどうかをチェックし(ステップS101)、全命令トレース実行中が0でないかどうかをチェックし(ステップS102)、全命令トレースデータを収集して、適宜2次記憶装置3に出力し(ステップS104)、シングルステップ割り込みを起こした命令のアドレスがユーザ空間かどうかをチェックし(ステップS105)、(一般に、全命令トレースの停止は、ユー

ザ空間で行うので)シングルステップ割り込みを起こした命令はPOPF Dかどうかをチェックする(ステップS106)が、(エミュレータ内でトレースを止めることは通常有り得ないので)NOとなり、ステップS200で(一般にオペレーティングシステム用情報10内の現スレッドアドレスと、予めユーザが指定した全命令トレースの終了用スレッドアドレスとが一致し、且つスタック50内の旧命令アドレス55と、予めユーザが指定した全命令トレースの終了用命令アドレスとが一致するかどうかをチェックすることにより)全命令トレースの終了が必要かのチェックを行い、ここでYESとなり、ステップS201へ進む。

【0045】ステップS201では、従来処理のトレース停止処理(一般にCPUが退避したスタック50内の旧状態フラグ53のトレースフラグ91をOFFとし、オペレーティングシステム用情報10内にあるすべてのスレッドのカーネルスタック上のトレースフラグ91をOFFとし、外部変数「全命令トレース実行中」に0を設定する処理)を行い、ステップS202で、外部変数「トレースON化累積カウンタ」が0かどうかをチェックする。0である場合は、従来の処理と同様に、ステップS205に行くことにより、全命令トレースを停止するが、この場合、外部変数「トレースON化累積カウンタ」は、1であるので、ステップS204へ行き、外部変数「全命令トレース仮停止」に1を設定し、ステップS205でレジスタ等の回復を行い、ステップS206でIRET命令を発行し、割り込み元へ戻る。

【0046】割り込み元に戻る時に、CPUは、自動的にスタック50内の旧状態フラグ53のトレースフラグ91をCPUのトレースフラグに設定し、その値は、ステップS201でOFFに設定されているので、戻った後、次の命令の実行直前等では、シングルステップ割り込みは発生しなくなる。しかし、エミュレータが、POPF D命令を実行すると、その時のユーザの状態フラグ60のトレースフラグ91は、上記ステップS109でONに設定したままであるので、該命令を実行した後に、CPUのトレースフラグはONに戻るので、その後の命令の実行直前にシングルステップ割り込みが発生し、再び、図4の全命令トレース用シングルステップ割り込み処理プログラムのステップS100に制御が渡る。その後、上記と同様に、ステップS100でレジスタ等の退避を行い、ステップS101で、外部変数「全命令トレース仮停止」が0であるかどうかをチェックするが、該変数には1が設定されているので、ステップS104の全命令トレースデータの収集等は行われず、ステップS110へ進む。

【0047】ステップS110では、(旧状態フラグ53の中のトレースフラグ91をOFFに設定することにより)CPUが退避したスタック50内のトレースフラグ91をOFFに設定し、外部変数「トレースON化累

積カウンタ」より1を減算し、この場合結果は0となるので、ステップS111においてYESとなり、ステップS203に進み、外部変数「全命令トレース仮停止」に0を設定し、レジスタ等の回復をステップS205で行い、ステップS206でIRET命令で割り込み元に戻る。その時、ステップS205で回復したトレースフラグはOFFに設定されているので、CPUのトレースフラグは0となり、以降、シングルスステップ割り込みは発生しなくなる。また、外部変数「全命令トレース仮停止」と「全命令トレース実行中」の内容は、共に0であるので、以降例えばシングルスステップ割り込みが発生しても、それは全命令トレースとは関係がないのでステップS102においてオペレーティングシステムのデバッグへ行くことになる。

【0048】

【発明の効果】以上説明したように、本発明によれば、オペレーティングシステム内に別のオペレーティングシステムのシステムコールの動作をユーザ空間上でエミュレートするエミュレータが搭載されている場合に、全命令トレース実行中に、エミュレータがCPUのトレースフラグをOFFにしようとしても、全命令トレース用シングルスステップ割り込み処理プログラムが、それを阻止して、且つ全命令トレースデータの収集を行い、また、全命令トレース仮停止中に、エミュレータがCPUのトレースフラグをONにしても、全命令トレース用シングルスステップ割り込み処理プログラムが、それをOFFに戻し、且つ、その時全命令トレースデータの収集を行わないので、従来処理では実現が不可能であったエミュレータにおける全命令トレースをすべて正しく行うことが可能となる。

【図面の簡単な説明】

【図1】本発明の一実施形態に係るエミュレータにおける全命令トレースの制御方法を実施する装置の構成を示

すブロック図である。

【図2】図1に示す実施形態における全命令トレース用シングルスステップ割り込み処理プログラムの走行時のスタックの構造を示す図である。

【図3】図1に示す実施形態における状態フラグの構造を示す図である。

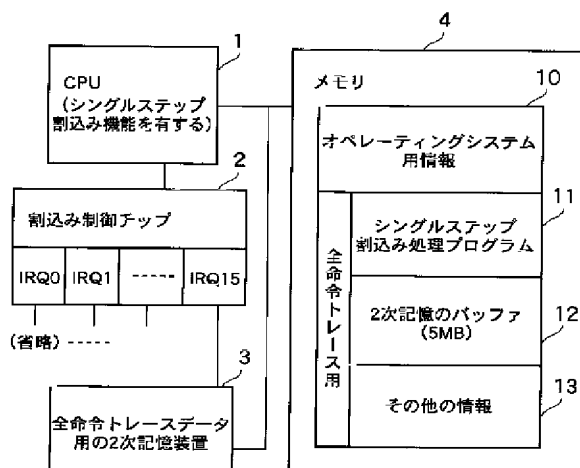
【図4】図1に示す実施形態の作用の一部を示すフローチャートである。

【図5】図1に示す実施形態の作用の図4に続く一部を示すフローチャートである。

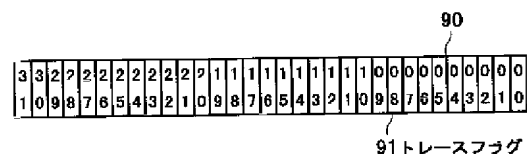
【符号の説明】

- 1 シングルスステップ割り込み機能を有するCPU
- 2 割り込み制御チップ
- 3 全命令トレースデータ用の2次記憶装置
- 4 メモリ
- 10 オペレーティングシステム用情報
- 11 全命令トレース用シングルスステップ割り込み処理プログラム
- 12 全命令トレース用2次記憶装置のバッファ
- 13 全命令トレース用のその他の情報
- 50 全命令トレース用シングルスステップ割り込み処理プログラムの走行時のスタック
- 51 旧スタックセグメント番号
- 52 旧スタックポインタ
- 53 旧状態フラグ
- 54 旧コードセグメント番号
- 55 旧命令アドレス
- 56 レジスタ情報群
- 60 ユーザの状態フラグ
- 70 スタックポインタ
- 90 状態フラグの構造
- 91 トレースフラグ

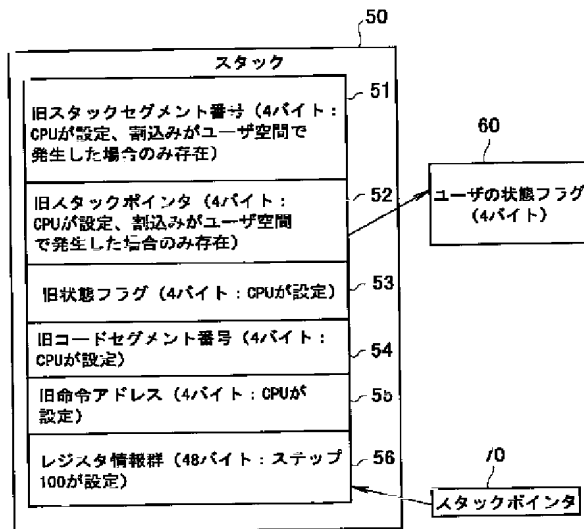
【図1】



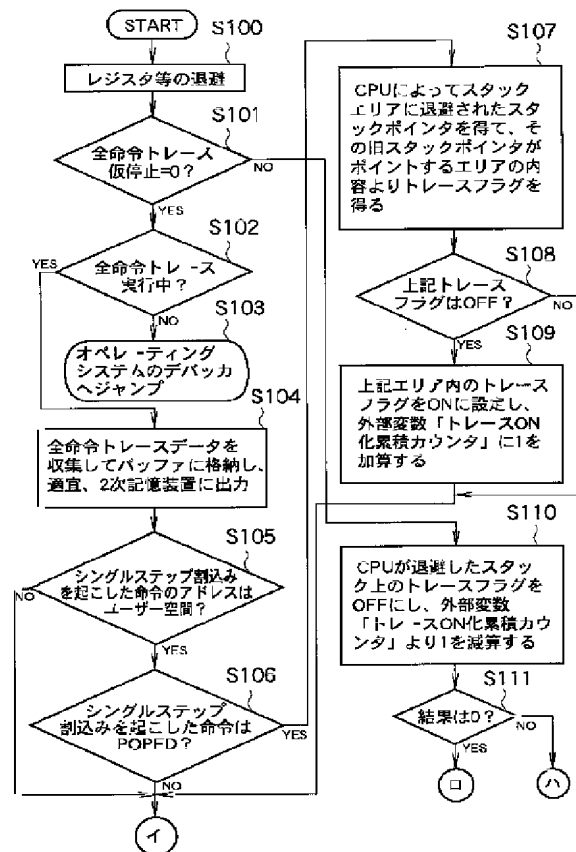
【図3】



【図2】



【図4】



【図5】

